A conceptual framework for predicting error in complex human-machine environments

Michael Freed and Roger Remington
NASA Ames Research Center

{mfreed, rremington@}mail.arc.nasa.gov

Abstract

We present a GOMS-MHP style model-based approach to the problem of predicting human *habit capture* errors. Habit captures occur when the model fails to allocate limited cognitive resources to retrieve task-relevant information from memory. Lacking the unretrieved information, decision mechanisms act in accordance with implicit default assumptions, resulting in error when relied upon assumptions prove incorrect. The model helps interface designers identify situations in which such failures are especially likely.

Introduction

Advances in our understanding of human cognition have not informed the design of complex human-machine systems to the extent possible. This results in part because the complexity of these systems poses a formidable challenge, and in part because the knowledge is not in form that can readily be applied in a design setting. Much of our knowledge of human capabilities and limitations comes from laboratory experiments using simple tasks and tight controls over extraneous variables. These controls are necessary to isolate mental operations of interest. However, it is hard to generalize the results to complex applied environments in which operators must plan the execution of multiple concurrent tasks in the face of considerable uncertainty. Under these conditions, no single mental operation determines behavior. Rather, it is necessary to understand how the diverse set of internal resources is managed to accomplish tasks. Even when certain known facts about human performance could be usefully applied in design, human-system designers would have difficulty locating those facts and understanding how they might apply to their specific problem. Such facts are

often buried in bulky sets of guidelines, whose rules themselves are often difficult to match to specific problems. Or, they are contained in scientific journals not easily comprehended by non-specialists. Handbooks can be useful, but still require the designer to know what information is needed and how to match the design requirements to the available data.

The introduction of GOMS modeling in conjunction with the Model Human Processor [Card84], made available a promising new methodology for dealing with complexity at a systems level. The Model Human Processor (MHP) provided researchers with a cognitive architecture whose resources and parameters constrained behavior while GOMS provided a formal method for procedure execution that enabled the representation of rules and procedures for selecting action in complex task domains [John94; Gray93].

Despite the success and widespread use of GOMS modeling, it has proven difficult to account for human error, or to handle the executive control needed to manage multiple tasks. These are significant shortcomings when modeling domains such as air traffic control where the coordination of multiple tasks is central and the concern with human error paramount. If we are to develop representations of human behavior that aid the designers of procedures and displays for air traffic control, we must deal directly with the source of human error in a dynamic, multitasking environment.

We have constructed a human operator model called APEX that is intended to help identify situations in which the design of equipment and procedures might inadvertently contribute to operator error [Freed97]. In keeping with the GOMS-MHP approach, APEX combines mechanisms for proceduralized task execution with a cognitive architecture that specifies resources. Our choice for a task execution model was driven by the demand for flexible scheduling of multiple tasks [Freed98a]. We replaced the GOMS component with a similar but more powerful procedure execution mechanism based on RAPs [Firby89]. Originally designed to enable robots to interleave and coordinate multiple tasks in dynamic, uncertain task environments, the RAP approach provides several important capabilities including:

?? continuous coordination of concurrent activities
?? diverse mechanisms for handling task interruption, task switching, and resumption
?? mechanisms needed to cope with uncertainty inherent in complex, dynamic environments
?? monitoring for and recovering from task failure

Our implementation of these capabilities is embedded within a human resource architecture that enforces human limitations on behavior [Freed98b]. Components of the architecture, each representing a perceptual, cognitive, or motor resource, are associated with limitations and parameters. For example, the vision component has a locus-of-

attention parameter. Execution can set this value to a single location in the current visual field. Because the visual component restricts access to visual information outside the selected location, agent performance depends on the effectiveness with which the locus-of-attention resource is allocated.

Using the execution module and resource architecture to simulate human behavior requires specifying domain-specific rules and procedures. Performance will depend on how those procedures use limited resources to carry out a task. Thus, the process of specifying procedures should be informed by an understanding of the strategies people use to manage limited resources. For example, people sometimes rely on written lists rather than faulty memory when shopping for groceries, or scan the market shelves for needed items, replacing a difficult memory task (recall) with an easier one (recognition). Such strategies become incorporated into people's routine procedures for carrying out a task, enabling them to circumvent limits that would otherwise affect performance [Salthouse91]. Modeling the effects of resource limitations on performance thus requires representing the procedural end-product of adaptation to routine tasks. While these adaptations are generally useful, they create the opportunity for error. We will discuss the role of such adaptations in producing a form of error called a habit capture and present a human operator model that incorporates this analysis to predict error in realistically complex environments.

Habit Capture Errors

Human error is an important concern in safety-critical work environments such as air traffic control. A survey of air traffic control related errors revealed that a high percentage of controller errors involve failures to carry out some intent, or failure to apply updated knowledge of the world in selecting an action. Errors involving failures to execute deferred intentions are examples of a class of memory phenomena referred to as prospective memory. Failures of prospective memory are common in daily life and include such errors as failing to take medication at prescribed times. Typical of prospective memory failures, operators often recognized their error shortly afterward. This suggests that at least some cases of prospective memory failures result not from a failure to successfully retrieve information, but a failure to make a retrieval attempt.

Our model ascribes such failures to initiate retrieval to the misallocation of limited resources during action selection. We illustrate the model using a class of prospective memory errors that we term **habit captures**. The signature of a habit capture error is the execution of a habitual action in place of an intended but non-routine action [Reason82]. A common example of such an error might be the failure to stop at the market on the way home. The intent is formed before leaving work, but cannot be carried out until the car reaches a specific turn-off. When this occurs, instead of exiting the highway at

the intended exit, the driver proceeds on the normal, habitual route.

Accounting for habit captures that result from failure to initiate a memory retrieval requires an understanding of when retrievals occur. By retrieval, we refer to memory access that requires the allocation of a limited capacity resource that model can only retrieve one item of information at a time [Carrier95]. The model assumes that no capacity-limited memory access is required for routine behaviors, which are encoded directly in procedures. For non-routine behaviors, the model must decide whether or not to allocate limited resources to retrieve the required information. It is this difference in the resource demands of routine and non-routine information that underlies the generation of habit capture errors.

Anomaly-driven retrieval

In making decisions about how to allocate resources, the model is guided by observed anomalies and internal goals. **Anomaly-driven memory retrievals** are initiated to explain unusual or ambiguous aspects of the current task environment. For example observing a basket of laundry in the middle of one's living room might trigger an attempt to locate an explanation in memory. People can take advantage of this aspect of human memory processing to provide timely reminders that help manage tasks. Thus, a person might intentionally place laundry in a conspicuous, atypical location as a reminder to do the wash. Similarly, people make use of unintended or incidental perceptual structure in the task environment to cue retrievals. For instance, if one were interrupted while bringing laundry to a washing machine, setting the laundry basket down might later serve to remind one to resume the task.

To simulate these anomaly-driven memory retrievals, our model assumes that people acquire expectations about the perceptual structure of their task environment and that they monitor these expectations in the normal course of carrying out a task. We further assume that when the environment regularly provides timely perceptual indicators that a memory retrieval is warranted, human decision-making processes adapt to take advantage of them. Such adaptations have been demonstrated in a variety of task domains; in some cases, people seem to use goal-driven retrieval in the early stage of learning a task, but gradually come to rely on perceptual indicators to initiate retrieval (see e.g. [Vera96]).

Learning to use environmental cues can be seen as an adaptive response to opportunity-costs associated with memory retrieval. Since only one memory retrieval attempt can be processed at a time, use of retrieval mechanisms for one task blocks or delays their use for all other tasks. Though they provide an efficient way to manage a limited resource, adaptations that rely on perceptual cues entail their own cost. In particular, when the usual cues are absent, reliance upon them may result in failure. For example, if someone removes a basket of laundry from the living room, its value in reminding a

person of their cleaning task will be undermined. More generally, *habit capture errors are especially likely when perceptual indicators normally present in the task environment are absent, and thus cannot trigger needed memory recall actions*.

It is helpful in analyzing such failures to contrast nominal behavior, in which a timely memory retrieval results in correct behavior, from error behavior in which no retrieval is initiated. In the latter case, a person behaves as if the unretrieved memory item had never been encoded. Decision-making processes can be described as operating under an implicit **default assumption** that some typical condition, opposite that implied by the memory item, holds in the current situation. In the example in which an intentionally placed laundry basket was removed and a failure to do laundry results, we could thus say that decision mechanisms implicitly assume that no intention to do laundry exists. The idea of a default assumption is useful in specifying what behavior is likely to follow when a relevant memory item is not retrieved. It also serves a useful practical purpose in explaining simulated behavior, allowing the simulation trace to make explicit reference to a critical non-event — i.e. the non-occurrence of a retrieval attempt.

Goal-driven retrieval

Goal-driven retrievals are initiated to acquire information for some active task. For example, one might query memory to determine where the car is parked when deciding where to exit a large office building. In our model [Freed98a], routine goal-driven behavior results from the execution of procedures, each represented as a set of **primitive** and **non-primitive** steps. Executing a primitive initiates activity in model resources, specifying simple actions such a gaze shift, utterance, or memory retrieval attempt.

Non-primitives specify a subgoal that, in many cases, can be accomplished by any of several alternative methods, each represented as a separate procedure. Executing a non-primitive requires selecting a method and then recursively executing each of its steps. Procedures often include **information acquisition steps** that satisfy information prerequisites for subsequent steps of the same procedure. For example, a procedure for getting home from work might include steps to acquire the location of one's car and then go to the specified location.

In many cases, information acquisition can be achieved by any of several alternative methods. Decision mechanisms can also forego explicit information acquisition, especially in highly routine tasks where the outcome of the acquisition process would tend to be some predictable value; instead, behavior conforms to the default assumption that this predictable value holds in the current situation. Thus, the exit path from one's office building can be selected by retrieving the car's current location from memory, visually scanning for the car out a window, or asking a companion. Alternately, one can simply leave by the usual exit without ever explicitly

considering the car's location. The implementation of our model treats reliance on a default explicitly — i.e. as another method for acquiring task-relevant information. To reflect its psychological status as an implicit rather than explicit event, the method of relying on a default takes no time and requires no limited resources.

In our model, which information acquisition method (including reliance on a default) is used in a particular instance depends on several factors. For this discussion, we will focus on one such factor. In particular, after learning of some unusual situation, we assume that a person will be less likely to rely on a default (that the usual situation holds) for some time thereafter. For instance, if a person parks his/her car somewhere other than the usual location, decision mechanisms that usually rely on the default location will have an increased likelihood of retrieving location from memory when exiting the building. We further assume that this likelihood decreases to the usual level over time, although time here is just a proxy for interference and other cognitive phenomena not currently represented in the model.

The likelihood of a habit capture error will thus depend partly on the amount of time since an unusual condition was observed and partly on the rate at which increased likelihood of retrieval declines. Our model handles the process of determining whether decision-mechanisms rely on a default or retrieve from memory in a simplified way. Whenever an intention or unusual condition is encoded (or retrieved), the model generates **bias**, causing any attempt to acquire information about that condition to avoid reliance on the default. Representations of bias are associated with a fixed **expiration interval**; when this interval has passed, the decision mechanisms revert to the usual method of determining an information acquisition method for the specified condition.

Like Anderson [1990], we assume that adaptive processes largely determine memory behavior, allowing experienced practitioners of a task to approximate optimal expiration interval values. An optimal interval weighs the risk of habit capture against the opportunity cost and time cost of retrieving a memory value that merely confirms the default. The ability of memory to approximate near-optimal intervals depends on experientially-derived knowledge of factors such as the expected duration D of a non-default condition, expected interval I between successive observations of a non-default (bias is refreshed each time the condition is observed), and expected risk of reducing performance at another task by blocking retrieval (opportunity cost). We approximate optimum expiration interval $EI = \min(D,I)$.

Since bias can be maintained by retrieving the non-default condition from memory, anomaly-driven mechanisms that use retrieval to explain non-default conditions can be used to support goal-driven retrieval. For instance, placing a written reminder that one's car is parked at an unusual location in a conspicuous place, and observing the reminder during the day,

increases the likelihood that decision mechanisms will explicitly consider the car's location when determining where to exit the building. *Such strategies will tend to fail (and result in habit capture errors) in the same conditions that the purely anomaly-driven strategies will fail — i.e. when needed perceptual support is absent.* We illustrate how our model simulates such an error in a hypothetical air traffic control scenario. Greater detail about the implementation of the model is given in [Freed98b].

Example air traffic control scenario

At a Terminal Radar Control center, one controller will often be assigned to the task of guiding planes through a region of airspace called an "approach sector" [Stein93]. This task involves contacting planes at various sector entry points and getting them lined up at a safe distance from one another on landing approach to a particular airport. Some airports have two parallel runways. In such cases, the controller will form planes up into two lines. Occasionally, a controller will be told that one of the two runways is closed and that all planes on approach to land must be directed to the remaining open runway. A controller's ability to direct planes exclusively to the open runway depends on remembering that the other runway is closed. How does the controller remember this important fact? Normally, the diversion of all inbound planes to the open runway produces an easily perceived reminder. In particular, the controller will detect only a single line of planes on approach to the airport, even though two lines (one to each runway) would normally be expected.

However, problems can arise in conditions of low workload. With few planes around, there is no visually distinct line of planes to either runway. Thus, the usual situation in which both runways are available is perceptually indistinguishable from the case of a single closed runway. The lack of perceptual support would then force the controller to rely on memory-driven retrieval and thus increase the chance of error.

Simulation and Implementation

In our air traffic controller simulation model, the arrival of a plane at a certain position in airspace (as observed on the radar display) causes the simulated controller to begin the task of selecting a destination runway for the target plane. We assume that for highly routine decisions such as runway selection, human controllers can be expected to know which factors to consider in making the decision and how to appropriately weight each factor. This knowledge is incorporated into the following *decision procedure*:

Procedure27: select runway for ?plane
1) *determine which runway has fewer planes on approach => ?factor1*
2) *det. which approach would be faster => ?factor2*
3) *det. which approach easier for me => ?factor3*

4) det. which runway safest for ?plane => ?factor4
5) det. left runway open? => ?factor5
6) det. right runway open? => ?factor6
7) compute-decision (factor1,factor2,..)

Generally, a decision procedure consists of n steps. The first (n-1) prescribe information acquisition tasks to evaluate potentially decision-relevant factors. The nth step runs a simple rule that selects from a fixed set of decision alternatives (left-runway or right-runway in this case) based on factor values.

Factor evaluation steps can typically be accomplished by any of several methods. In this example, the controller could determine the status of the left runway by retrieving information from memory, asking another controller, or by assuming the most likely condition – i.e. that the runway is open. Since runway closures are rare and memory retrieval is expensive, we assume that a typical controller will rely on the default unless transient bias promotes a more effortful alternative.

In the described scenario, the simulated controller hears that the left runway is closed. Interpretation mechanisms cause a propositional representation of this fact to be encoded in memory. The encoding event generates bias according to the following rule:

IF (closed ?runway) is encoded in memory
THEN bias procedure-27, step5. (expire in 10 min)

Consequently, procedure execution mechanisms will be biased against relying on the default value when carrying out step5 of procedure27 for the next ten minutes — i.e. the availability of the left runway will be verified rather than assumed when selecting a runway for an approaching plane.

Eventually, the initial bias expires. To select a runway for a newly arrived plane, the controller will once again consider only the default assumption. Other factors then determine which runway is selected. For example, the controller may choose to direct a heavy plane to the longer left runway which, in normal circumstances, would allow the plane an easier and safer landing. With the left runway closed, actions following from this decision result in error.

Avoiding error requires maintaining appropriate bias. In a variation of the described scenario in which no error occurs, visually perceived reminders of the runway closure cause bias to be periodically renewed. In particular, whenever visual attention mechanisms attend to plane icons on an approach path to the airport, interpretation mechanisms note the absence of a line of planes to the left runway and signal an expectation failure on the basis of the following rule:

IF I am visually attending to left approach path, and
visual group of plane icons not detected

THEN signal-anomaly: (absent plane-group left)

In general, whenever an expectation failure occurs, a task to explain the observed anomaly is initiated. The first step in such a task is to try to match the anomaly to a known explanation-pattern (XP) [Schank86]. A match results in a task to verify the explanatory hypothesis provided by the XP.

Explanation-pattern
Anomaly: (absent plane-group ?left-or-right)
Candidate Explanation: (closed runway ?runway)
To verify: retrieve from memory (closed runway ?runway)

In principle, verifying a hypothesis could involve mental and physical actions of any kind. In this case, the contents of working memory are sufficient to prove or disprove the explanation; the anomalous absence of planes on approach to the left runway is explained as a result of the left runway's closure.

Bias renewal occurs whenever the working memory item that originally produced the bias is reencoded or retrieved. Thus, retrieving (closed runway left) triggers the bias generation rule just as if the proposition had been encoded for the first time. Thus, the unusual arrangement of planes on the radar scope acts as a constant reminder, preventing the controller from reverting to the use of its default assumption and thereby preventing error.

Aiding user interface design

By helping to simulate such scenarios, the model can direct an interface designer's attention to potential design-facilitated errors that might otherwise be overlooked. Moreover, the model's ability to make explicit how such errors might occur can help indicate the best way to refine an interface. For example, one of the difficulties in designing a radar display is balancing the need to present a large volume of information against the need to keep the display uncluttered. In this case, by showing how the error results from low traffic conditions, the simulation indicates a clever fix for the problem: use an icon to explicitly represent runway closures, but only display the icon in low plane-load conditions when it is most needed and produces the least clutter

Conclusion

We have presented a GOMS-MHP style approach to the problem of predicting human habit capture errors in the domain of air traffic control. Our model assumes that people manage limited memory retrieval resources by taking advantage of perceptual indicators that a retrieval is warranted, and by incorporating knowledge about when retrievals should occur into routine procedures. Habit captures occur when decision mechanisms fail to retrieve intentions or knowledge of unusual conditions; lacking the unretrieved information, decision mechanisms act in accordance with

implicit default assumptions resulting in error. The model helps to identify situations in which such errors are especially likely. Interface designers can then use this information to reduce the likelihood of error.

References

Anderson, J.R. (1990) *The Adaptive Character of Thought*. Lawrence Earlbaum Associates.

Baddeley, A.D. (1986) *Working Memory*. Oxford University Press.

Card, S.K., Moran, T.P., and Newell, A. (1983). *The psychology of human-computer interaction*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Carrier, L.M. and Pashler, H. (1995). Attentional limitations in memory retrieval. *Journal of Experimental Psychology: Learning, Memory, & Cognition*, **21**, 1339-1348.

R.J. Firby. (1989) *Adaptive execution in complex dynamic worlds*. Ph.D. thesis, Yale University.

Freed, M. and Remington, R. (1997) Managing decision resources in plan execution. In *Proceedings of the Fifteenth Joint Conference on Artificial Intelligence*, Nagoya, Japan.

Freed (1998a) Managing multiple tasks in complex, dynamic environments. In *Proceedings of the 1998 National Conference on Artificial Intelligence*, Madison, WI.

Freed (1998b) *Simulating human performance in complex, dynamic environments*. Ph.D. thesis. Northwestern Univ.

Gray, W. D., John, B. E., Atwood, M.E. (1993). Project Ernestine: Validating a GOMS Analysis for Predicting and Explaining Real-World Task Performance. *Human Computer Interaction*, **8**, 237-309.

[John94] John, B.E. and Kieras, D.E. (1994). *The GOMS Family of Analysis Techniques: Tools for Design and Evaluation*. Carnegie Mellon University, School of Computer Science, TR CMU-CS-94-181.

Kieras, D.E. and Meyer, D.E. (1997) An overview of the EPIC architecture for cognition and performance with application to human-computer interaction. *Human-Computer Interaction*, in press.

Reason, J.T. (1990). *Human Error*. Cambridge, UK: Cambridge University Press.

Reason, J.T. and Mycielska, K. (1982). *Absent-minded? The psychology of mental lapses and everyday errors*. Englewood Cliffs, NJ: Prentice-Hall.

Salthouse, T.A. (1991). Expertise as the circumvention of human processing limitations. In Ericsson, K.A. and Smith J.A. (Eds.), *Toward a general theory of expertise* (pp. 286-300), Cambridge.

Schank, R.C. (1986) *Explanation Patterns*, Lawrence Earlbaum Associates, Hillsadale, N.J.

Stein, Earl S. and Garland, Daniel. (1993). Air traffic controller working memory: considerations in air traffic control tactical operations. FAA technical report DOT/FAA/CT-TN93/37.

Vera, A.H. and Lewis, R.L. (1996) Dissociating performance from learning: an empirical evaluation of a computational model. In *Proceedings of the eighteenth annual conference of the cognitive science society*, Lawrence Earlbaum Associates, 409-414, 1996.